FINDING THE SHORTEST CYCLE THROUGH k SPECIFIED NODES

by

Gilbert Laporte**
and
Yves Nobert***

** Ecole des Hautes Etudes Commerciales de Montréal
5255, avenue Decelles, Montréal, CANADA H3T 1V6

*** Département des Sciences administratives
Université du Québec à Montréal
1495, rue Saint-Denis, Montréal, CANADA H3C 3P8

The authors are grateful to the Canadian Natural Sciences and Engineering Research Council (grant A4747) and to the Québec Government (F.C.A.C. grant 83EQ0428), for their financial support; they also thank Martin Desrochers who executed some of the computational tests.

Université de Montréal

Centre de recherche sur les transports - Publication #307

Ecole des Hautes Etudes Commerciales
Les Cahiers du Gerad G-83-07

Avril 1983

Dépôt légal - 3e trimestre 1983

ABSTRACT

Let G = (N,E) be an undirected graph where N is the set of nodes and E, the set of edges. Let C be a symmetrical distance matrix defined on N^2 and K, a subset of N. This paper considers the problem of determining the shortest cycle containing each node of K exactly once and each node of N-K at most once. The problem is formulated as an integer linear program and solved by three different algorithms. Numerical results are reported.

TITRE: DETERMINATION DU CYCLE LE PLUS COURT PASSANT PAR k NOEUDS SPECIFIES

RESUME

Soit G = (N,E) un graphe non orienté où N est l'ensemble des noeuds et E, l'ensemble des arêtes. Soit C une matrice de distances non symétrique définie sur N^2 et K, un sous-ensemble de N. Dans cet article, on considère le problème de la détermination du cycle de longueur minimale contenant chaque noeud de K exactement une fois et chaque noeud de N-K au moins une fois. Le problème est formulé au moyen de la programmation linéaire en nombres entiers et résolu par trois algorithmes différents. On présente des résultats numériques.

1. INTRODUCTION

Let G = (N,E) be a complete graph where N = {1,...,n} is the set of nodes and E, the set of (undirected) edges; let C be a matrix whose elements c_{ij} represent distances between pairs of nodes i and j. Since G is undirected, it follows that C is symmetrical, i.e. $c_{ij} = c_{ji}$ for all edges (i,j). (In this paper, c_{ji} must be interpreted as c_{ij} whenever $i < j$). C satisfies the triangle inequality if and only if $c_{ik} + c_{kj} \geq c_{ij}$ for all i,j,k. The matrix $C' = (c'_{ij})$ derived from C by defining c'_{ij} as the shortest distance between i and j always satisfies the triangle inequality.

We consider the following problems associated with G.

Problem 0 : (TSP: Symmetrical Travelling Salesman Problem [1]): find the shortest (undirected) cycle containing each node of N exactly once. (Such a cycle is said to be Hamiltonian).

Problem 1 : (SCP: Shortest Cycle Problem through k specified nodes [4]): let K = {1,...,k} be a set of "specified" nodes ($k \leq n$); find the shortest (undirected) cycle containing each node of K exactly once.

Problem 2 : (GTSP: Generalized Travelling Salesman Problem through r sets of nodes [6]): let $S_1,\ldots,S_r$ be subsets of N such that $S_1 \cup \ldots \cup S_r = N$. The problem consists of finding the shortest Hamiltonian cycle containing at least one node in each of $S_1,\ldots,S_r$.

Problem 3 : (SCCP: Shortest Complete Cycle Problem [10]): find the shortest cycle containing each node of N at least once.

The TSP is well known and has been studied extensively in the Operational Research literature (see [1] and also [7] for a recent partial survey). This paper focuses on the SCP which has not received as much attention as the TSP. It is however a very challenging problem from a theoretical point of view and is more general than the TSP since it includes it as a special case (K = N). The GTSP is also a generalization of the TSP and, as we shall see, contains the SCP as a special case. Fi-

nally, the SCCP constitutes a relaxation of the TSP; it has been treated by an efficient constraint relaxation algorithm in [10]. In the following section, we outline further relationships between these four problems.

2. VARIANTS OF THE SCP

Several variants of the SCP can be derived according to the nature of C and to the restrictions imposed on the degree of nodes belonging to N - K. These are summarized in table 1. We first observe that problems where C satisfies the triangle inequality or where the degree of nodes in N - K is equal to 2 are only of limited interest since they always reduce to a TSP on N or on K. The last three problems in the second column of the table deserve some comment.

Degree of nodes in N-K greater than or equal to 2

This problem lies somewhere between the TSP (K = N) and the SCCP (K = ϕ). It is therefore only of limited interest and can be handled by an algorithm similar to that proposed for the SCCP [10].

Degree of nodes in N-K non-specified

Dreyfus [2] has shown that this problem reduces to a TSP on K where the distance matrix is equal to C'.

Degree of nodes in N-K less than or equal to 2

This appears to be the most difficult case. Apart from the paper by Dreyfus, we have identified two other studies on this problem in the Operational Research literature. The first one by Saksena and Kumar [11] contains a serious mistake which was pointed out by Dreyfus [2] and should be disregarded. The second one, by Ibaraki [4], treats a slightly different problem (that of finding a shortest path instead of a shortest cycle) and deals with the asymmetrical case. The author suggests two different algorithms. In the first one, the problem is formulated as an

TABLE 1

Variants of the SCP

Degree of nodes in N - K	C satisfies the triangle inequality	C does not satisfy the triangle inequality
= 2	TSP on N (distance matrix C)	TSP on N (distance matrix C)
> 2	TSP on N (distance matrix C)	Hybrid problem between the TSP on N and the SCCP on N.
≤ 2	TSP on K (distance matrix C restricted to K^2)	Problem treated in this paper
Non-specified	TSP on K (distance matrix C restricted to K^2)	TSP on K (distance matrix C' first defined on N^2 and then restricted to K^2)

integer linear program containing degree constraints, integrality constraints and subtour elimination constraints. The algorithm proposed follows the lines of an earlier algorithm by Martin [8] for the TSP and later exploited by several other researchers (see [7]): a relaxed problem containing only the degree constraints is first solved; integrality is obtained by generating Gomory cutting planes [3] and illegal subtours are eliminated as in Dantzig et al. [1]. The second algorithm takes advantage of the fact that the first relaxed problem is an assignment problem

and is therefore unimodular. Integrality is preserved throughout the algorithm while illegal subtours are eliminated through the scheme proposed by Shapiro [12]. No computational results are reported for the first algorithm; problems involving 21 and 31 nodes were solved by the second method. Ibaraki's paper also includes a description of the GTSP but does not outline the relationship between the GTSP and the SCP.

Much progress has been made in recent years in the use of constraint relaxation algorithms for the (symmetrical) TSP and for many of its variants [7]. These studies generally confirm the efficiency of this type of algorithm. It has also become clear that it is not in general efficient to transform symmetrical problems into asymmetrical problems (by duplicating the edges) and to solve them by using an algorithm devised for asymmetrical problems: the number of subtours involving only two nodes tends to be prohibitive and leads to a proliferation of branches in the search tree. These observations lead us to believe that it may be preferable to devise an algorithm enabling symmetrical problems to be solved directly and taking advantage of some recent developments in constraint relaxation methods.

3. FORMULATION

In this formulation of the SCP, x_{ij} and y_i are binary variables indicating whether or not edge (i,j) (i<j) and, respectively, node i of N-K, are included in the optimal cycle; we also define $\bar{T} = N-T$.

(P1) **minimize** $\sum_{i<j} c_{ij} x_{ij}$

subject to

(1) $\sum_{\ell<i} x_{\ell i} + \sum_{i<j} x_{ij} = 2$ $\qquad (i \in K)$

$$(2) \quad \sum_{\ell<i} x_{\ell i} + \sum_{i<j} x_{ij} = 2y_i \qquad (i \in N - K)$$

$$(3) \quad \sum_{i,j \in T} x_{ij} \leq |T| - 1$$

$$(3') \quad \sum_{\substack{i \in T, j \in \bar{T} \\ \text{or } i \in \bar{T}, j \in T}} x_{ij} \geq 2$$

$$(T \subset N,\ T \cap K \neq \phi,\ T \cap K \neq K)$$

$$(4) \quad y_i = 0, 1 \qquad (i \in N - K)$$

$$(5) \quad x_{ij} = 0, 1 \qquad (i,j \in N,\ i<j)$$

This formulation is similar to that of Dantzig et al. for the TSP [1] and reduces to it when K = N. It will not therefore be necessary to comment on it in detail here. In the TSP, it was not necessary to include the two types of subtour elimination constraints ((3) and (3')) since these were equivalent when all nodes had a degree equal to 2. However, as we shall see later, it may be necessary, in some circumstances, to generate subtour elimination constraints while some of the y_i's are still fractional: in such cases, (3) may be over-satisfied while (3') provides a valid cut.

Before we describe our algorithm for the SCP, note that this problem can be viewed as a special case of the GTSP and hence can be handled directly through the GTSP algorithm presented in [6]. This can be done by partitioning N in the following way:

$$S_1 = \{1\} \cup (N - K)$$

$$S_i = \{i\} \qquad (i \in K - \{1\})$$

In order to ensure that node 1 will belong to the optimal solution, one replaces c_{1j} by $c_{1j} - M$ ($j \in N - \{1\}$; $M \gg 0$). But this transformation offers no computational advantage over (P1) and will not be used as such.

4. ALGORITHM

Here are the main steps of the algorithm used to solve (P1):

Step 0 : Use a heuristic procedure in order to obtain an initial solution.

Step 1 : Solve a relaxed version of (P1) including the degree constraints ((1) and (2)) and the upper bounds on the variables.

Repeat Steps 2 and 3 until an optimal solution has been identified. In the "STRAIGHT" algorithm, start with Step 2; in the "REVERSE" algorithm, start with Step 3.

Step 2 : Obtain an integer solution by branch and bound or by generating Gomory cutting planes [3].

Step 3 : Introduce violated subtour elimination constraints ((3) and/or (3')).

If Gomory cutting planes are used to reach integrality, the algorithm terminates when an integer solution containing no illegal subtours has been obtained; in the branch and bound version, the process stops when all branches of the search tree have been explored.

Three versions of this algorithm were tested.

Version 1: "STRAIGHT" algorithm with branch and bound

The branch and bound routine used was the Land and Powell BB program [5]. One type (3) constraint was introduced for each illegal subtour. It would have been valid to use type (3') constraints, but these contain on the average a larger number of non-zero coefficients and are therefore less attractive from a computational point of view.

Version 2: "STRAIGHT" algorithm with Gomory cuts

As in [9] and in other studies (see [7]), only one Gomory cut was generated for each fractional solution. (The choice of the "best" cut is fully described in [9]). Ineffective cuts are eliminated from the program once the value of the objective function has increased. Type (3) constraints are again used to eliminate illegal subtours: they always remain in the program.

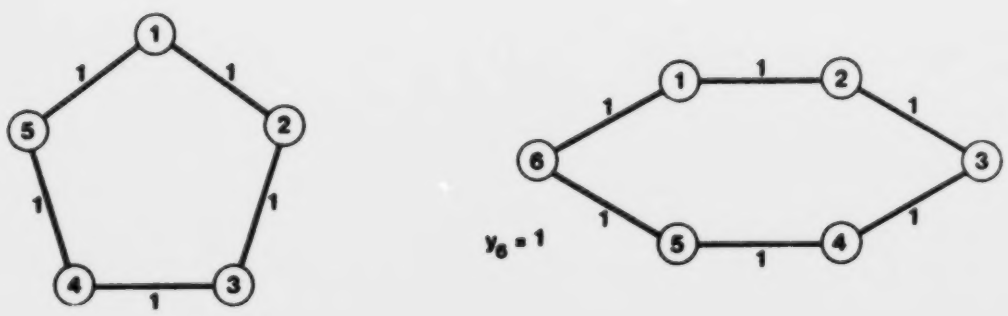
Version 3: "REVERSE" algorithm with Gomory cuts

In order to illustrate how Gomory cuts and subtour elimination constraints are used in this version, let us consider a problem where n = 10 and k = 5. Solutions satisfying degree constraints are made up of "connected components" and, possibly, of isolated nodes belonging to N - K. (A connected component is a connected subgraph of G having non-zero x_{ij}'s associated with all its edges). Note that no connected component with all its nodes contained in N - K will be created.

There are four types of connected components.

(i) Components containing all nodes of K and no fractional variables. They constitute an optimal solution (see Figure 1).

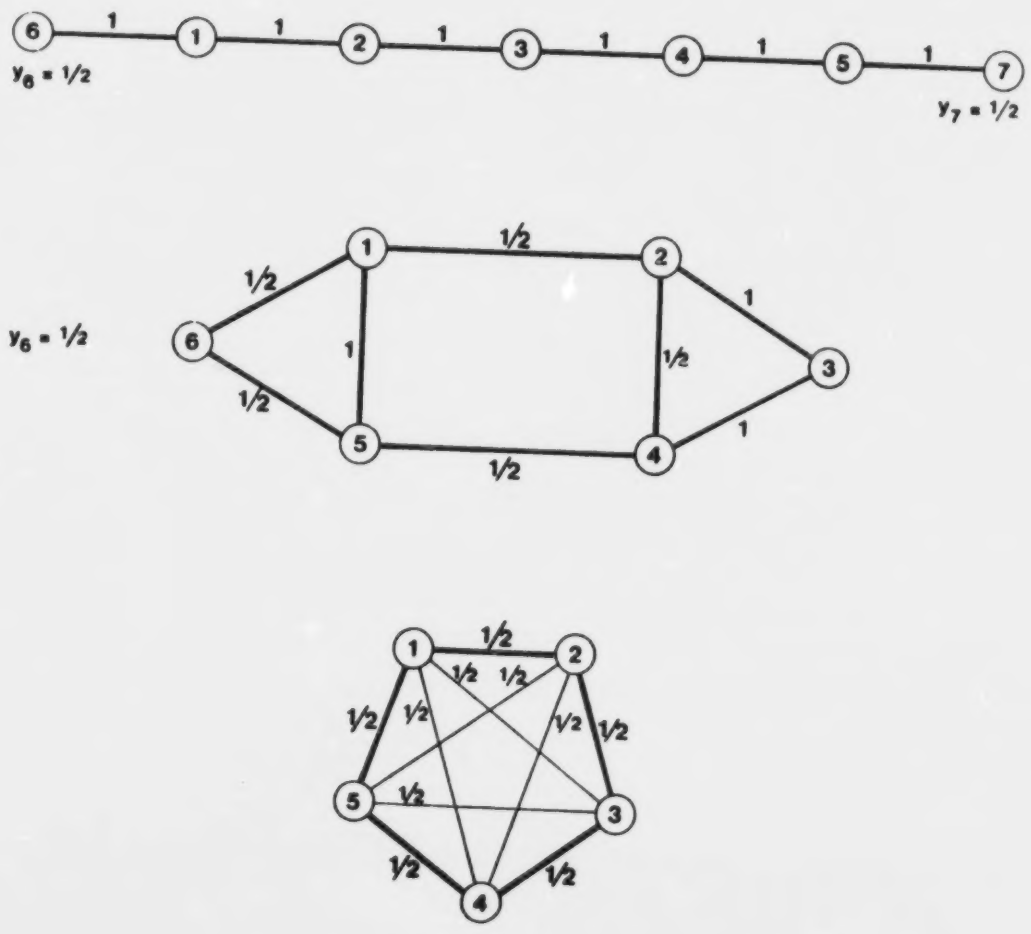
Figure 1
All nodes of K and no fractional variables



(ii) Components containing all nodes of K and some fractional variables. These components are eliminated through the introduction of a Gomory cut since they do not violate any constraint of type (3) or (3'): these constraints are indeed never generated when $T \cap K = K$ (see Figure 2).

Figure 2

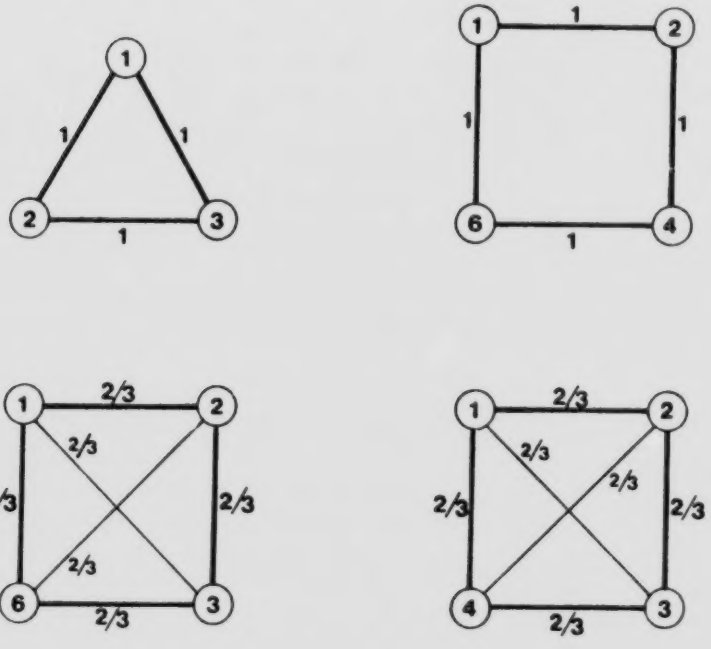
All nodes of K and some fractional variables



(iii) Components containing only some nodes of K and no fractional y_i variables. In these components, all degree constraints are satisfied and therefore, the sum of all x_{ij}'s is equal to the number of nodes. One constraint of type (3) is therefore violated and can be generated to eliminate the solution (see Figure 3).

Figure 3

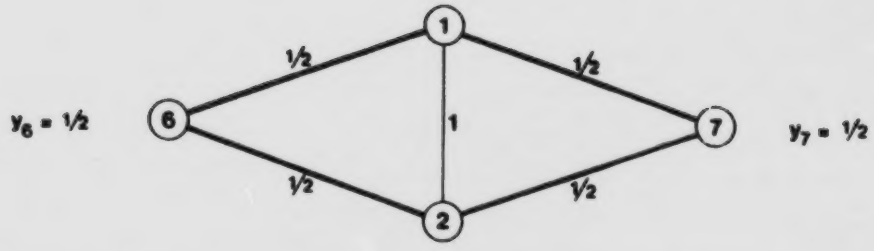
Some nodes of K and no fractional y_i variables



(iv) Components containing only some nodes of K and at least one fractional y_i variable. These components do not always violate constraints (3) and are destroyed through the introduction of a type (3') constraint (see Figure 4).

Figure 4

Some nodes of K and at least one fractional y_i variable



5. COMPUTATIONAL RESULTS

The three versions of the algorithm were tested on a series of problems whose c_{ij}'s were generated randomly on the interval [0,100]. Problems were solved on the University of Montreal CDC Cyber 173 computer. A time limit of 300 seconds was imposed for the solution of each problem.

TABLE 2

"STRAIGHT" algorithm with branch and bound

n	k	maximum level of search tree	number of subtour elimination constraints (3)	number of iterations	time (seconds)
20	5	8	1.7	253	13.32
	10	5	2.7	116	5.89
	15	3.3	3	130.3	6.97
	20	0.7	2.7	62.3	3.27
30	7	19.7	9	1634	190.12
	15	11	5	479	50.80
	22	5.7	5	242	28.08
	30	0.7	2	84	8.99
40	10	15.5	7	975	214.70
	20	9.7	4	459.7	82.94
	30	4	3.3	232	39.17
	40	1	5.7	134.7	25.88
50	37	8.3	5	309.3	100.64
	50	1	2	153.7	41.75

Results are reported in tables 2 and 3. These tables indicate average results for the successful problems; 3 problems of each kind were attempted and solved successfully with version 1; with the other two versions, 5 problems were attempted, but some failed. It can be observed that version 1 is more reliable than versions 2 and 3. However, the Gomory cuts methods are on the average faster over the successful problems.

TABLE 3

Algorithms with Gomory cuts

Type	n	k	Number of problems solved	Number of Gomory cuts	Number of subtour elimination constraints (3)	Number of subtour elimination constraints (3')	Number of iterations	Time (seconds)
VERSION 2: "STRAIGHT"	20	5	4	8	0	-	42.25	4.6
		10	5	10.2	2	-	63.4	6.86
		15	5	5.2	16	-	51.6	4
	30	7	1	14	2	-	61	13.1
		15	2	10.5	0	-	66.5	10.94
		22	5	4.6	1.6	-	74.8	11.27
	40	30	1	9	2	-	180	26.8
VERSION 3: "REVERSE"	20	5	3	3.66	.33	4	46.6	4.57
		10	5	9.4	1	2.6	75	8.91
		15	5	5.5	1.6	.8	46.2	2.97
	30	23	3	3	2.3	23	69.3	10.6

REFERENCES

[1] DANTZIG, G.B., FULKERSON, D.R., JOHNSON, S.M.: "Solution of a Large Scale Travelling Salesman Problem", Operations Research, vol.2, pp.393-410, 1954.

[2] DREYFUS, S.E.: "An Appraisal of Some Shortest-Path Algorithms", Operations Research, vol.17, pp.395-412, 1969.

[3] GOMORY, R.: "An Algorithm for Integer Solutions to Linear Programs", Recent Advances in Mathematical Programming, pp.269-302, McGraw-Hill, 1963.

[4] IBARAKI, T.: "Shortest Path Problems Visiting Specified Nodes", Electronics and Communications in Japan, vol.53-A, pp.10-18, 1970.

[5] LAND, A.H., POWELL, S.: "FORTRAN Codes for Mathematical Programming: Linear, Quadratic and Discrete", Wiley, 249 pp., 1973.

[6] LAPORTE, G., NOBERT, Y.: "Generalized Travelling Salesman through n Sets of Nodes: an Integer Programming Approach", to appear in INFOR, 1983.

[7] LAPORTE, G., NOBERT, Y.: "Algorithmes de relaxation de contraintes pour le problème du voyageur de commerce symétrique et ses extensions", to appear in Les Annales des Sciences mathématiques du Québec, 1983.

[8] MARTIN, G.: "Solving the Travelling Salesman Problem by Integer Linear Programming", Working Paper, Control Data Corporation, 1966.

[9] MILIOTIS, P.: "Using Cutting Planes to Solve the Symmetric Travelling Salesman Problem", Mathematical Programming, vol.15, pp.177-188, 1978.

[10] MILIOTIS, P., LAPORTE, G., NOBERT, Y.: "Computational Comparison of Two Methods for Finding the Shortest Complete Cycle or Circuit in a Graph", RAIRO (recherche opérationnelle), vol.15, pp.233-239, 1981.

[11] SAKSENA, J.P., KUMAR, S.: "The Routing Problem with "K" Specified Nodes", Operations Research, vol.14, pp.909-913, 1966.

[12] SHAPIRO, D.M.: "Algorithms for the Solution of the Optimal Cost and Bottleneck Travelling Salesman Problems". Ph.D. Thesis, Washington University, 1966.